HW_ML_Revise_2

"Kittipoom Bank"

2023-08-08

Load Library

```
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v lubridate 1.9.2     v tibble    3.2.1
## v purrr     1.0.1     v tidyr     1.3.0
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(readxl)
library(ggplot2)
library(dplyr)
library(lmtest)
```

```
## Loading required package: zoo
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

Import excel file

```
df_Y2016 <- read_excel("House Price India.xlsx", sheet=1)
```

Select Parameter for study

```
lm_model_checkoverall <- train(Price ~ . ,
                         data=df_Y2016,
                         method="lm")

lm_model_checkoverall
```

```
## Linear Regression
##
## 14620 samples
##    22 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 14620, 14620, 14620, 14620, 14620, 14620, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   185564.1  0.7427495  103667.2
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
lm_model_checkoverall$finalModel
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
##
## Coefficients:
##                                     (Intercept)
##                                       1.567e+11
##                                               id
##                                      -2.320e+01
##                                            Date
##                                       9.021e+00
##                      `\\`number of bedrooms\\``
##                                      -3.529e+04
##                    `\\`number of bathrooms\\``
##                                       2.590e+04
##                           `\\`living area\\``
##                                       1.198e+02
##                              `\\`lot area\\``
##                                      -1.841e-01
##                        `\\`number of floors\\``
##                                      -2.290e+04
##                     `\\`waterfront present\\``
##                                       5.430e+05
##                        `\\`number of views\\``
##                                       3.683e+04
##                 `\\`condition of the house\\``
##                                       1.265e+04
##                     `\\`grade of the house\\``
##                                       5.546e+04
## `\\`Area of the house(excluding basement)\\``
```

```
##                                   4.626e+01
##                  `\\`Area of the basement\\``
##                                          NA
##                            `\\`Built Year\\``
##                                  -1.582e+03
##                       `\\`Renovation Year\\``
##                                   1.152e+01
##                           `\\`Postal Code\\``
##                                   1.062e+03
##                                    Lattitude
##                                   1.739e+05
##                                    Longitude
##                                  -9.890e+04
##                              living_area_renov
##                                  -1.949e+01
##                                 lot_area_renov
##                                  -3.647e-01
##               `\\`Number of schools nearby\\``
##                                   1.630e+03
##             `\\`Distance from the airport\\``
##                                  -1.072e+02
```

```
varImp(lm_model_checkoverall)
```

```
## lm variable importance
##
##   only 20 most important variables shown (out of 21)
##
##                                                Overall
## id                                            100.0000
## `\\`waterfront present\\``                     57.8019
## `\\`living area\\``                            50.2656
## `\\`grade of the house\\``                     43.9231
## `\\`Built Year\\``                             39.1810
## `\\`number of bedrooms\\``                     34.4984
## `\\`number of views\\``                        30.9693
## `\\`Postal Code\\``                            24.2488
## Lattitude                                      23.9589
## `\\`Area of the house(excluding basement)\\``  18.9100
## Longitude                                      14.5189
## `\\`number of bathrooms\\``                    13.9620
## `\\`number of floors\\``                       10.9579
## living_area_renov                               9.5075
## `\\`condition of the house\\``                  9.2087
## lot_area_renov                                  8.1308
## `\\`lot area\\``                                5.8154
## `\\`Renovation Year\\``                         5.2118
## `\\`Number of schools nearby\\``                0.9854
## `\\`Distance from the airport\\``               0.4768
```

Select top 5 of Significant Parameter

waterfront present / living area / grade of the house/ Built Year / number of bedrooms

Even if, id is first effect but actually in my view show not related. [So not select]

Simplify Parameter Name

```
study_df <- df_Y2016 %>% select (
                           "waterfront" = "waterfront present",
                           "living_area" = "living area" ,
                           "grade_house" = "grade of the house",
                           "built_year" = "Built Year",
                           "bedrooms" = "number of bedrooms",
                           "Price"
)
```

Add Column Log Price

```
study_df <- study_df %>% mutate(log_price = log(Price))
```

1: split data 80% train, 20% test

```
split_data <- function(df) {
  set.seed(42)
  n <- nrow(df)
  train_id <- sample(1:n, size = 0.8*n)
  train_df <- df[train_id, ]
  test_df <- df[-train_id, ]
  return( list(training = train_df,
               testing = test_df) )
}

prep_data <- split_data(study_df)
train_df <- prep_data[[1]]
test_df <- prep_data[[2]]
```

2.1: train model [Normal Method]

```
lm_model <- train(Price ~ waterfront + living_area + grade_house + built_year + bedrooms ,
                  data=train_df,
                  method="lm")

lm_model
```

```
## Linear Regression
##
## 11696 samples
##     5 predictor
```

```
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 11696, 11696, 11696, 11696, 11696, 11696, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   210023.1  0.6561228  139685.3
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

2.2: train model [Take Log Method]

```
lm_model_log <- train(log_price ~ waterfront + living_area + grade_house + built_year + bedrooms ,
                      data=train_df,
                      method="lm")

lm_model_log
```

```
## Linear Regression
##
## 11696 samples
##     5 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 11696, 11696, 11696, 11696, 11696, 11696, ...
## Resampling results:
##
##   RMSE       Rsquared   MAE
##   0.3133998  0.6448899  0.2504367
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

3.1: score model [Normal Method]

```
p <- predict(lm_model, newdata=test_df)
```

3.2: score model [Take Log Method]

```
p_log <- predict(lm_model_log, newdata=test_df)
```

4.1: evaluate model [Normal Method]

mean absolute error

```
(mae <- mean(abs(p - test_df$Price)))
```

```
## [1] 143786.7
```

root mean square error

```
    (rmse <- sqrt(mean((p - test_df$Price)**2)))
```

```
## [1] 246417.3
```

4.2: evaluate model [Take Log Method] (Test Data)

mean absolute error

```
   (mae_log_test = mean(abs(exp(p_log) - exp(test_df$log_price))))
```

```
## [1] 147813.6
```

root mean square error

```
    (rmse_log_test = sqrt( mean((exp(p_log) - exp(test_df$log_price))**2)))
```

```
## [1] 417188.5
```

4.3: evaluate model [Take Log Method] (Train Data)

```
p_train <- predict(lm_model_log, newdata=train_df)
```

mean absolute error

```
   (mae_log_train = mean(abs(exp(p_train) - exp(train_df$log_price))))
```

```
## [1] 133266.5
```

root mean square error

```
    (rmse_log_train = sqrt( mean((exp(p_train) - exp(train_df$log_price))**2)))
```

```
## [1] 211072.6
```

5: Compare Result

Normal Method with 5 Parameter

```
print(lm_model)
```

```
## Linear Regression
##
## 11696 samples
##     5 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 11696, 11696, 11696, 11696, 11696, 11696, ...
## Resampling results:
##
```

```
##   RMSE      Rsquared   MAE
##   210023.1  0.6561228  139685.3
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
print(mae)
```

```
## [1] 143786.7
```

```
print(rmse)
```

```
## [1] 246417.3
```

Take Log Method with 5 Parameter

```
print(lm_model_log)
```

```
## Linear Regression
##
## 11696 samples
##     5 predictor
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 11696, 11696, 11696, 11696, 11696, 11696, ...
## Resampling results:
##
##   RMSE       Rsquared   MAE
##   0.3133998  0.6448899  0.2504367
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

```
print(mae_log_test)
```

```
## [1] 147813.6
```

```
print(rmse_log_test)
```

```
## [1] 417188.5
```

```
print(mae_log_train)
```

```
## [1] 133266.5
```

```
print(rmse_log_train)
```

```
## [1] 211072.6
```